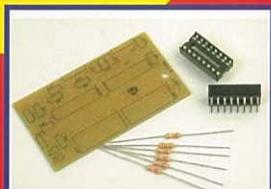
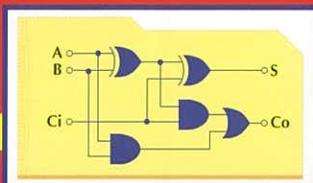
impara l'elettronica digitale

...e costruisci il tuo **LABORATORIO DIGITALE**

6,90 €

9

9 771824 363008 41009

Peruzzo & C.

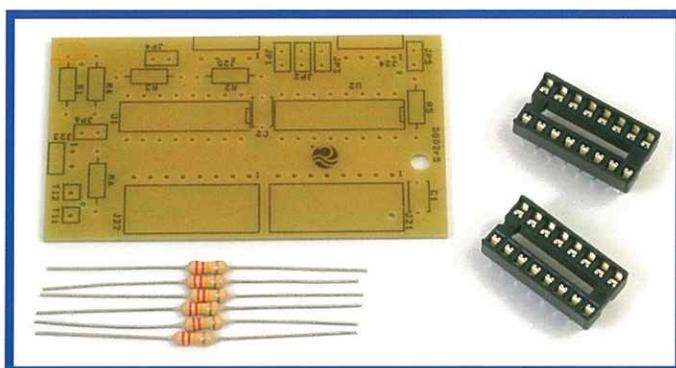
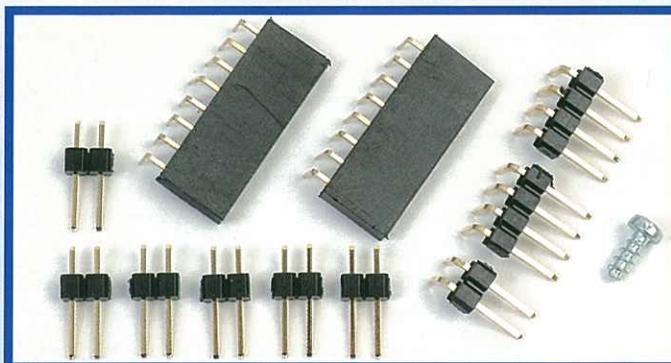
Direttore responsabile:
ALBERTO PERUZZO
Direttore Grandi Opere:
GIORGIO VERCELLINI
Consulenza tecnica e traduzioni:
CONSULCOMP S.n.c.
Pianificazione tecnica
LEONARDO PITTON

Direzione, Redazione, Amministrazione: viale Ercole Marelli 165, Tel. 02/242021, 20099 Sesto San Giovanni (Mi). Pubblicazione settimanale. Registrazione del Tribunale di Monza n. 1738 del 26/05/2004. Spedizione in abbonamento postale gr. II/70; autorizzazione delle Poste di Milano n. 163464 del 13/2/1963. Stampa: Staroffset s.r.l., Cernusco S/N (MI). Distribuzione SO.DI.P. S.p.A., Cinisello Balsamo (MI).

"ELETTRONICA DIGITALE"
si compone di
70 fascicoli settimanali
da suddividere
in 2 raccoglitori.

RICHIESTA DI NUMERI ARRETRATI. Per ulteriori informazioni, telefonare dal lunedì al venerdì ore 9.30-12.30 all'ufficio arretrati tel. 02/242021. Se vi mancano dei fascicoli o dei raccoglitori per completare l'opera, e non li trovate presso il vostro edicolante, potrete riceverli a domicilio rivolgendovi direttamente alla casa editrice. Basterà compilare e spedire un bollettino di conto corrente postale a PERUZZO & C. s.r.l., Ufficio Arretrati, viale Marelli 165, 20099 Sesto San Giovanni (MI). Il nostro numero di c/c postale è 42980201. L'importo da versare sarà pari al prezzo dei fascicoli o dei raccoglitori richiesti, più le spese di spedizione € 3,10 per pacco. Qualora il numero dei fascicoli o dei raccoglitori sia tale da superare il prezzo globale di € 25,82 e non superiore a € 51,65, l'invio avverrà per pacco assicurato e le spese di spedizione ammonteranno a € 6,20. La spesa sarà di € 9,81 da € 51,65 a € 103,29; di € 12,39 da € 103,29 a € 154,94; di € 14,98 da € 154,94 a € 206,58; di € 16,53 da € 206,58 in su. Attenzione: ai fascicoli arretrati, trascorse dodici settimane dalla loro distribuzione in edicola, viene applicato un sovrapprezzo di € 0,52, che andrà pertanto aggiunto all'importo da pagare. Non vengono effettuate spedizioni contrassegno. Gli arretrati di fascicoli e raccoglitori saranno disponibili per un anno dal completamento dell'opera. IMPORTANTE: è assolutamente necessario specificare sul bollettino di c/c postale, nello spazio riservato alla causale del versamento, il titolo dell'opera nonché il numero dei fascicoli e dei raccoglitori che volete ricevere.

IN REGALO in questo fascicolo

1 Circuito stampato DG02RS
2 Zoccoli DIL 16 pin
6 Resistenze 220 K 5% 1/4 W

IN REGALO nel prossimo fascicolo

2 Connettori femmina da c.s. a 90° 8x1
1 Connettore maschio da c.s. a 90° 2x1
2 Connettori maschi da c.s. a 90° 4x1
6 Connettori maschi da c.s. dritti a 2 pin
1 Vite

COME RACCOGLIERE E SUDDIVIDERE L'OPERA NELLE 4 SEZIONI

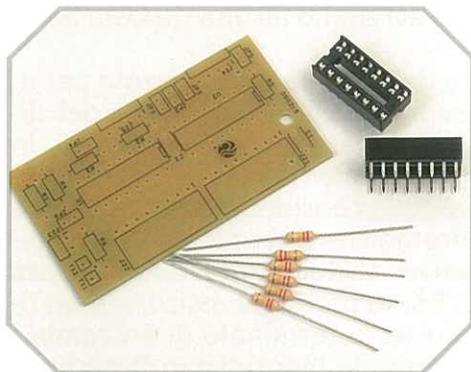
L'Opera è composta da 4 sezioni identificabili dalle fasce colorate, come indicato sotto. Le schede di ciascun fascicolo andranno suddivise nelle sezioni indicate e raccolte nell'apposito raccoglitore, che troverai presto in edicola. Per il momento, ti consigliamo di suddividere le sezioni in altrettante cartellette, in attesa di poterle collocare nel raccoglitore. A prima vista, alcuni numeri di pagina ti potranno sembrare ripetuti o sbagliati. Non è così: ciascuno fa parte di sezioni differenti e rispecchia l'ordine secondo cui raccogliere le schede.
Per eventuali domande di tipo tecnico scrivere al seguente indirizzo e-mail:
elettronicadigitale@microrobots.it

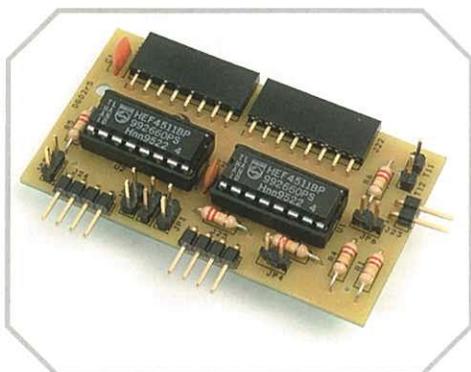
Hardware Montaggio e prove del laboratorio

Digitale di base Esercizi con i circuiti digitali

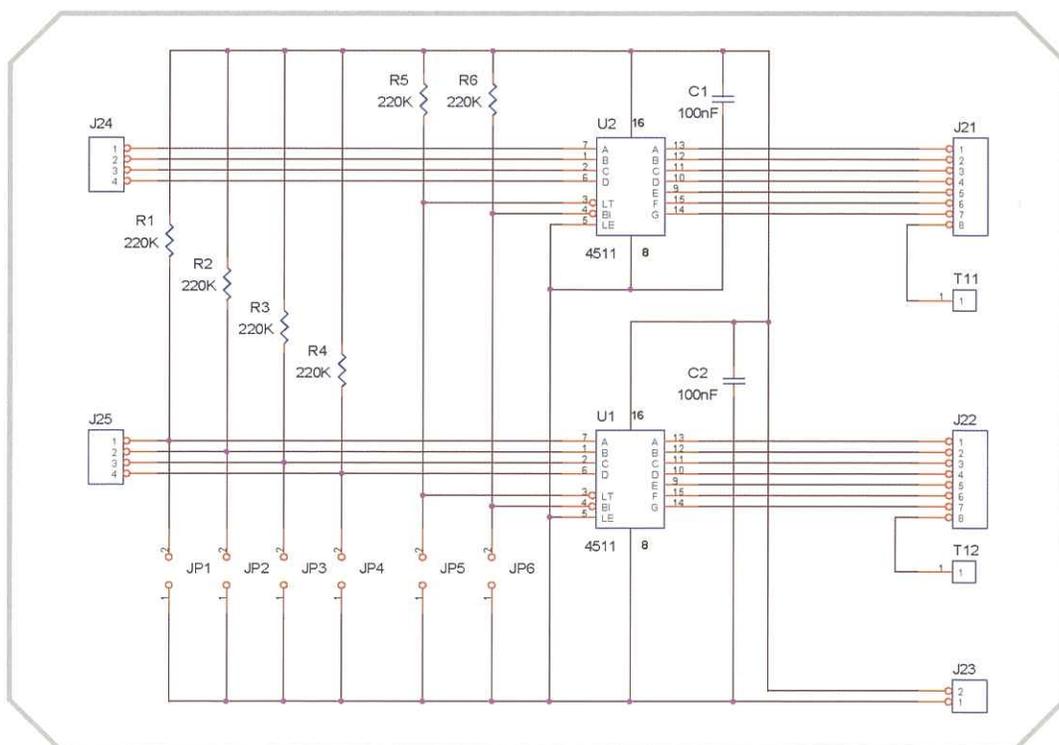
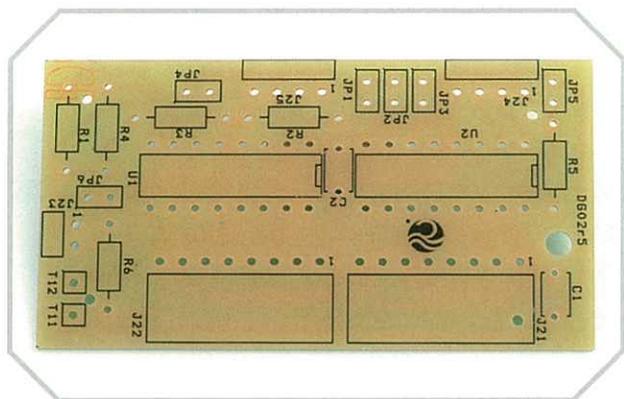
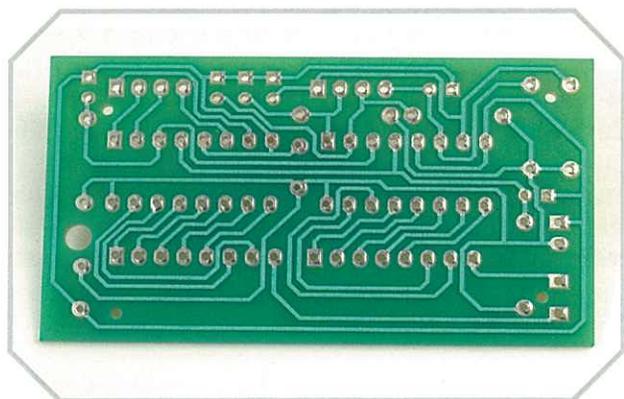
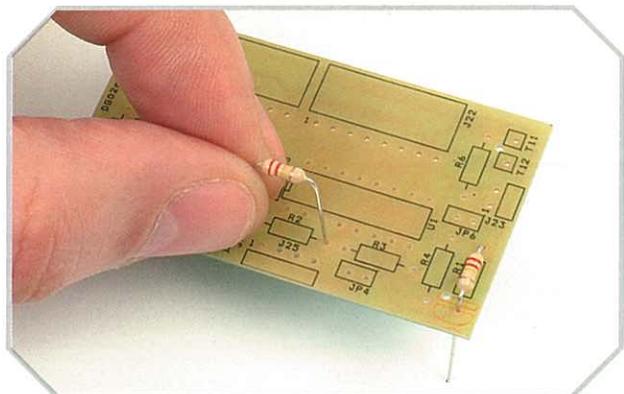
Digitale avanzato Esercizi con i circuiti sequenziali

Microcontroller Esercizi con i microcontroller

Driver per display

Circuito stampato DG02 e alcuni dei suoi componenti.

Iniziamo il montaggio del circuito stampato DG02, che supporta il driver per i display da 7 segmenti già montati sulla scheda DG01. Monteremo anche i due zoccoli da 16 pin e le 6 resistenze da 220 K. Questo è il secondo circuito stampato necessario per costruire il contatore completo che verrà utilizzato in diversi esercizi pratici.

Aspetto del circuito terminato.

Il circuito

I principali componenti di questo circuito stampato, quando sarà completo, sono due circuiti integrati driver con ingresso in codice binario da 4 bit per display e uscita a 7 segmenti. Ciascuno di questi circuiti integrati ha 4 terminali di ingresso che corrispondono ai 4 bit con cui si rappresentano in binario le combinazioni che si associano ai numeri in decimale dallo 0 al 9; nel caso applicassimo i codici rimanenti, ovvero quelli corrispondenti ai valori decimali dal 10 al 15, questi sarebbero

Schema elettrico del driver DG02.

Circuito stampato.

Lato saldature.

Inizio del montaggio.

ignorati e non avremmo alcuna risposta all'uscita.

Ogni uscita dalla A alla G è utilizzata per illuminare il segmento corrispondente del display a cui è collegata e che è installato, in questo caso, sul circuito DG01.

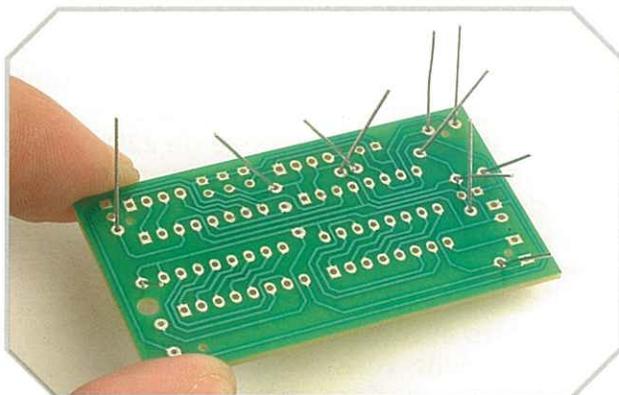
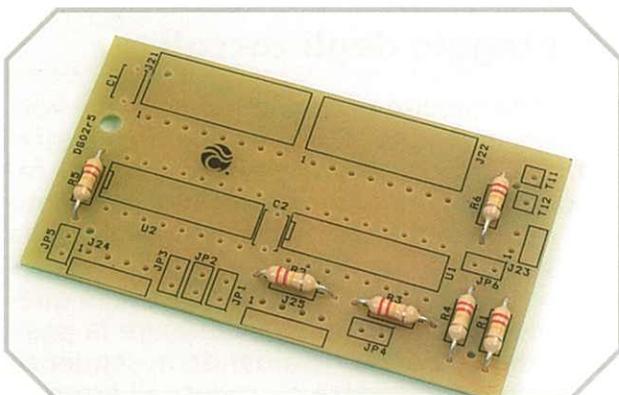
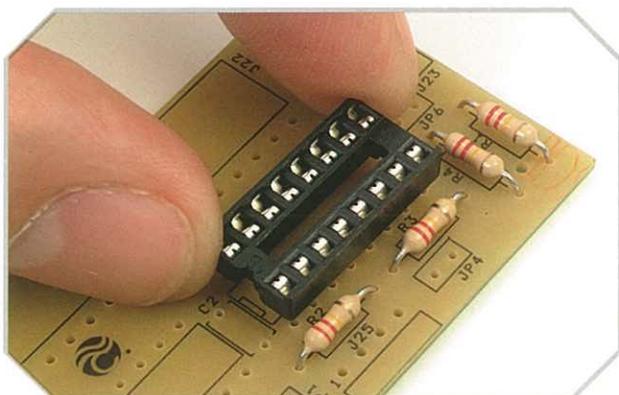
Se prendiamo in considerazione lo schema elettrico vedremo che il terminale LT è collegato al positivo dell'alimentazione tramite la resistenza R5, in modo da assicurare un livello alto su questo terminale di entrambi i circuiti, mantenendo l'ingresso in stato inattivo. Questo terminale può essere collegato al negativo dell'alimentazione inserendo un ponticello tra i terminali di JP5 in modo che i 7 segmenti dei display, se sono collegati, si illumineranno, quindi il terminale servirà per verificare se le connessioni fra il display e i segmenti sono corrette e l'adeguato funzionamento del display.

Il pin BI di entrambi i circuiti si utilizza per la funzione contraria; quando si attiva, ovvero si uniscono fra loro i due terminali di JP6, si spengono tutti i segmenti. Non bisogna inserire entrambi i ponticelli JP5 e JP6 contemporaneamente.

Ponti di prova

Se osserviamo attentamente lo schema elettrico di questo circuito vedremo quattro resistenze con sigle che vanno da R1 a R4 le quali mantengono a livello alto i quattro bit di ingresso; stiamo cioè applicando 1111 che corrisponde al 15, quindi nessun segmento del display, controllato in questo caso da U1, si illuminerà.

Noteremo inoltre quattro gruppi da due terminali siglati da JP1 fino a JP4, in modo che quando si uniscono tra loro i terminali di ognuno di questi connettori si applica un 1 all'ingresso corrispondente. Ad esempio se poniamo un ponticello su JP2 e un altro su JP3, il codice applicato all'ingresso del driver sarà 1001 che corrisponde a 9 e questa cifra si illuminerà sul display. Come si può vedere i ponticelli ci permetteranno di eseguire delle prove in codice binario e verificarne il risultato sul display.

Questi ponticelli e resistenze sono elementi aggiuntivi che normalmente non si utilizzano nei circuiti driver.

Le resistenze si fissano allargando i loro terminali.

Resistenze montate.

Dobbiamo rispettare l'orientamento degli zoccoli.

Terminali ausiliari

I pin siglati come T11 e T12 sono utilizzati per montare terminali ausiliari per controllare l'illuminazione del punto di ogni display; questo punto è un LED addizionale che si può illuminare applicando la tensione positiva di alimentazione su questi terminali. Il display a più digit si utilizza per indicare il punto decimale.

Condensatori

I condensatori C1 e C2 filtrano la corrente di alimentazione ed evitano la possibile influenza di disturbi che possono sovrapporsi all'alimentazione. Come vedremo il circuito può funzionare anche senza questi condensatori quando è alimentato a batterie. Anche se non sono strettamente necessari conviene montarli, cosa che faremo più avanti.

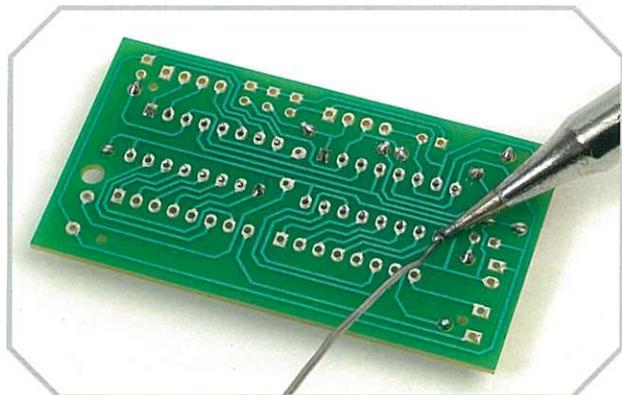
Terminali di uscita

I terminali di uscita del circuito sono raggruppati su due connettori, uno per display. I terminali J22 corrispondono al display delle unità e quelli di J21 al display delle decine. Sette di questi terminali sono utilizzati per illuminare un solo segmento del display ciascuno e l'ottavo si utilizza per illuminare il punto.

Terminali di ingresso

Questo circuito ha due connettori di ingresso, J24 e J25, l'ultimo dei quali si utilizza per applicare il codice a quattro bit che contiene l'informazione del numero rappresentato sul display delle unità e l'altro è dedicato al codice corrispondente al display delle decine.

Terminali di alimentazione

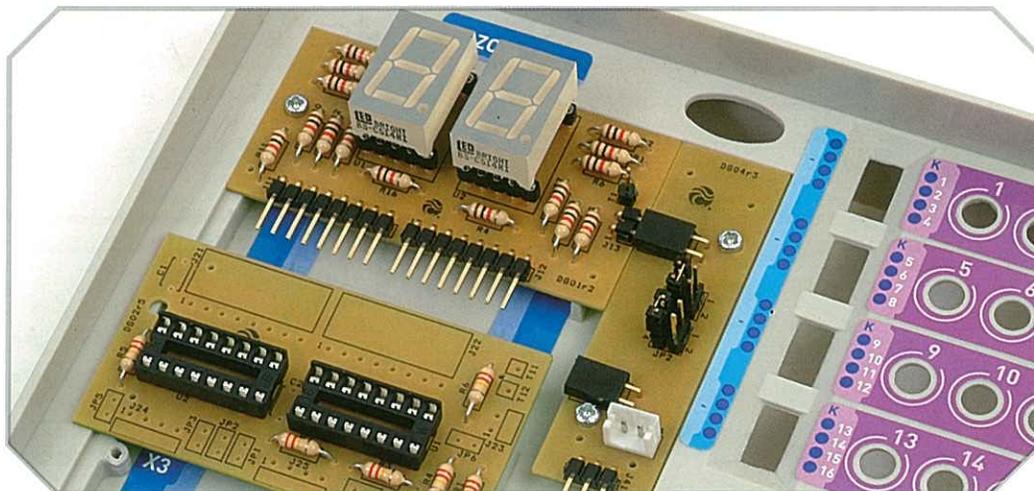
Questo circuito riceve l'alimentazione tramite il connettore a due terminali J23, il terminale 1 è il negativo dell'alimentazione e il terminale 2 il positivo. Questo connettore riceve l'alimentazione tramite il connettore J43 del circuito di alimentazione 1, la cui sigla è DG04. Questo circuito, grazie alla tecnologia CMOS della serie 4000, può essere alimentato indistintamente alle tensioni disponibili sul laboratorio: 4,5, 5 o 9 volt.

Saldatura dei due terminali opposti per fissare l'integrato.

Circuito stampato con i componenti disponibili.

Montaggio

Il montaggio di questo circuito inizia, come al solito, con i componenti più bassi, in questo caso sono le sei resistenze da 220 K (rosso, rosso, giallo). Inseriremo ognuna di queste resistenze nelle posizioni indicate da R1 a R6 e le salderemo. Per facilitare questa operazione bisogna invertire la scheda, ma prima di fare questo, conviene separare in modo opposto al contenitore del componente i terminali delle resistenze che fuoriescono dalla parte posteriore del circuito stampato, come possiamo vedere nelle figure, per evitare che queste cadano e poter realizzare le saldature con le resistenze collocate nella posizione ottimale. Dopo averle saldate taglieremo la parte di reoforo in eccesso di ogni resistenza.

Montaggio degli zoccoli

In questo circuito verranno montati due zoccoli che, in seguito, ospiteranno i circuiti integrati. Gli zoccoli devono essere montati facendo coincidere la loro tacca di riferimento con quella disegnata sulla serigrafia della scheda. Per saldare questi componenti inizieremo dai due pin agli angoli opposti, in questo modo potremo ancora cambiare la posizione dello zoccolo riscaldando in sequenza i due pin e correggere gli eventuali errori di montaggio. Quando saremo sicuri della posizione potremo eseguire le saldature rimanenti.

PCB vicino alla sua posizione definitiva.

Somma e sottrazione

Il cervello umano si abitua a realizzare operazioni complesse compresa la somma, al punto da realizzarle quasi inconsciamente, dimenticando praticamente il processo di apprendimento molte volte duro e difficile, che abbiamo sostenuto durante i primi anni di scuola. Il cervello si abitua e "lotta" contro i cambiamenti, per questo quando si cambia la base ed è necessario sommare, "questa lotta" genera un'apparente difficoltà.

Somma

La somma in binario fondamentalmente è uguale alla somma in decimale, tuttavia dovremo memorizzare solamente quattro dati, in quanto, avendo unicamente due simboli, non abbiamo che quattro combinazioni possibili e due di queste sono uguali.

Prima di iniziare a spiegare la somma in binario, però, ricordiamo come si esegue una somma in decimale con un semplice esempio: abbiamo due sommandi, il 3 e il 7, il risultato è 10, nulla di sorprendente; se osserviamo attentamente notiamo che il risultato della somma delle unità è zero, quindi è stato necessario aggiungere una cifra a sinistra per poter rappresentare l'uno.

Vediamo un altro esempio, in questo caso vogliamo sommare 17 e 23; sommando le unità otterremo 10, lo zero verrà scritto nel posto delle unità e l'uno si somma alla colonna delle decine che sarà 1 + 2 + 1, con il risultato di 4, ottenendo 40 come risultato finale della somma.

Somma in binario

La somma si realizza bit a bit, se un numero ha diversi bit si inizia la somma con il meno significativo, ovvero dal lato destro. La somma di due bit è semplice, basta osservare la tabella. Ci sono solamente quattro possibili combinazioni, 0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1 e, in ultimo, 1 + 1 = 0, però con un 1 di riporto e il risultato è 10 (uno zero), cioè 2 in decimale, non dieci. Questo 1 che ci obbliga ad aggiungere un BIT alla sinistra deve essere sommato al secondo BIT, nel caso di una somma con più bit. Come avete potuto vedere il procedimento è lo stesso della somma in decimale.

B	A	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Tabella della verità della somma di 2 bit. A e B addendi. C, (Carry) riporto. S, somma.

0	0	1	1
+0	+1	+0	+1
0	1	1	10

Riporto (Carry)

Somma di 2 bit. Quando i due addendi sono 1 si produce un riporto.

Somma di diversi bit

Normalmente vengono rappresentate con 4, 8 o più cifre. In questo caso si somma prima il bit meno significativo (LSB) di ogni numero, e si ottiene il primo bit del risultato, anch'esso è il meno significativo, ovvero quello di destra. Se c'è riporto (in inglese carry) si aggiunge alla somma dei due bit successivi, ottenendo come risultato il secondo bit; se c'è nuovamente riporto si aggiunge alla somma dei bit successivi, e così via fino a terminare i bit; un eventuale riporto alla fine si chiama overflow, dato che è necessario disporre di un bit in più per poter rappresentare il dato ottenuto.

Somma di 4 bit.

La sottrazione è una somma, ma al sottraendo è stato applicato il complemento a 1 e sommato 1 al risultato.

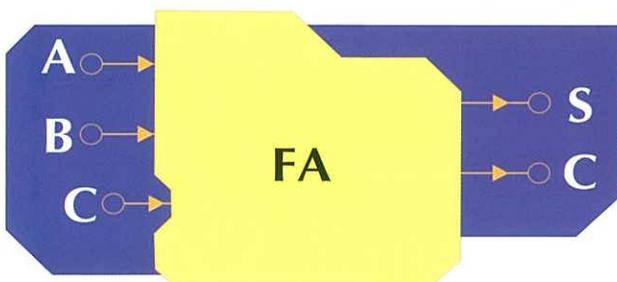
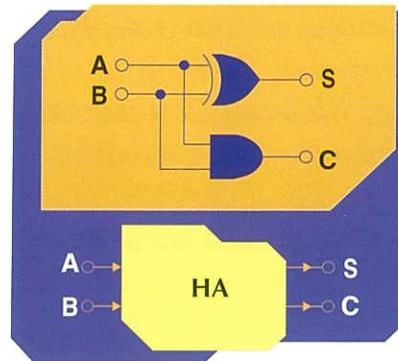
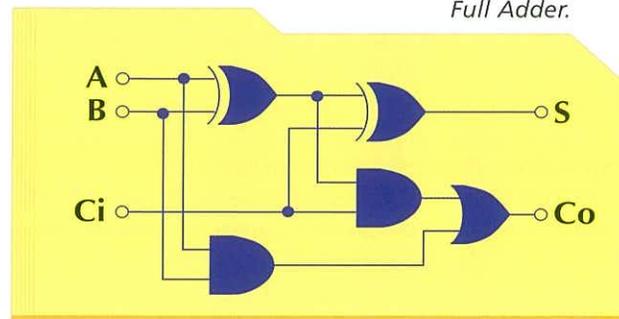
Complemento a uno

Il complemento a uno di un numero in binario si ottiene cambiando i valori uno con zero e viceversa. È molto utile perché frequentemente lo si utilizza per realizzare operazioni matematiche. Si ottiene molto facilmente applicando a ogni bit un inverter, per ottenere direttamente il complemento a uno.

Complemento a due

Il complemento a due di un numero si ottiene sommando uno al complemento a uno. Ha un interesse prevalentemente teorico, in quanto è utilizzato nella sottrazione, tuttavia nei circuiti disponibili attualmente sul mercato, salvo alcune eccezioni piuttosto datate, è più facile e comodo ottenere il complemento a uno e sommare un uno.

Sottrazione

La sottrazione in binario si realizza a partire dalla somma. Ciò che si fa è cambiare il sottraendo e realizzare operazioni di somma. Esistono diversi procedimenti, vediamo qual è il più utilizzato.

Sommatore completo, Full Adder, FA, con ingresso di riporto.

Semisommatore Half Adder, HA.

Schema di un sommatore completo, Full Adder.

Supponiamo di avere due numeri che rappresentiamo come A e B e con cui vogliamo realizzare la sottrazione A – B. In questo caso ciò che realmente si fa con i circuiti è quanto segue: si somma A + (complemento a due di B), anche se si potrebbe fare A + (complemento a uno di B9) + 1.

Circuiti

Il circuito che somma due bit e dà come risultato la somma e il riporto si chiama semisommatore (in inglese Half Adder, HA).

Il sommatore completo (Full Adder, FA) è uguale a quello precedente, però ha un terminale per accettare il riporto originato nella somma del bit precedente, in modo che il circuito possa sommare dei bit a partire dal secondo se si ottiene da riporto di un bit precedente. È anche possibile concatenare diversi sommatori per sommare numeri di molti bit.

Di solito è possibile reperire sul mercato sommatori a quattro bit, dato che sono abbastanza utilizzati, attualmente però con l'arrivo dei circuiti a logica programmabile e i microcontroller, questi circuiti iniziano a diventare inutili dal momento che possono essere implementati con questi dispositivi più complessi.

MPLAB

MPLAB è uno strumento software per il progetto e lo sviluppo di applicazioni con microcontroller PIC della ditta Arizona Microchip Technology (AMT). Contiene tutte le utilità necessarie per la realizzazione di qualsiasi progetto, permette la scrittura del file sorgente in linguaggio assembler, la compilazione e la simulazione su display. Inoltre permette l'esecuzione del programma in modalità passo a passo per vedere come evolvono in modo reale i registri interni, la memoria RAM e/o EEPROM e la memoria di programma, mano a mano che si eseguono le istruzioni.

Installazione del programma

Il CD allegato contiene la versione 5.70 di MPLAB. Se carichiamo il CD nel nostro lettore e selezioniamo l'opzione "Programmi", potremo vedere apparire l'icona di installazione mostrata nella figura.

Icona di installazione di MPLAB.

Quando eseguiamo il file di installazione appare una videata di benvenuto che ci chiede se desideriamo installare l'applicazione.

Se accettiamo con il pulsante Next il programma risponde con una nuova videata dove dovremo accettare le condizioni del contratto di licenza per l'installazione. Se accettiamo le condizioni "I agree" e clicchiamo nuovamente su Next, apparirà un'altra videata in cui potremo scegliere i moduli dell'applicazione che desideriamo installare.

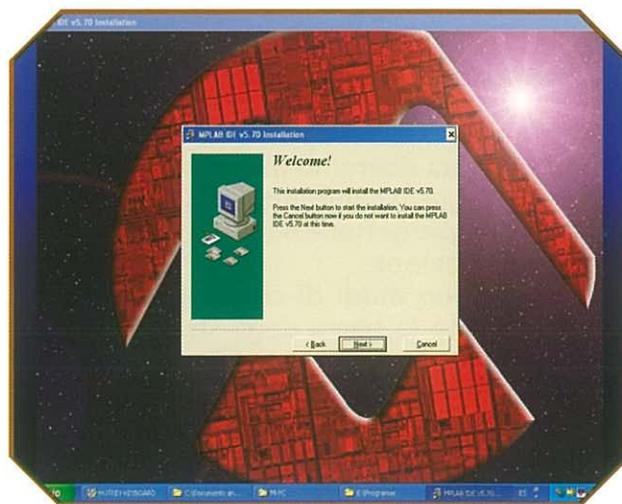
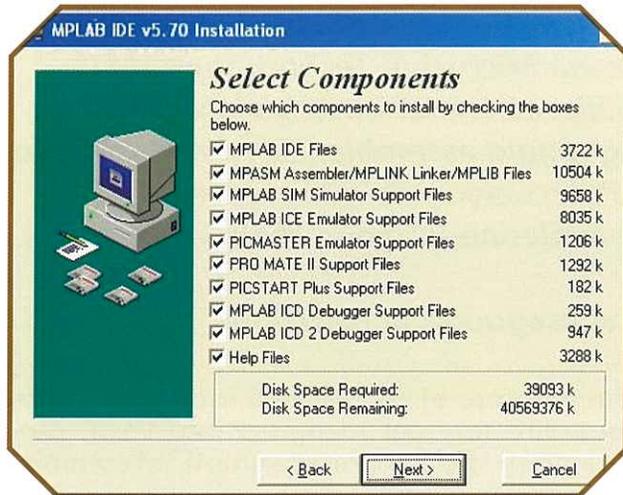
Dato che lo spazio richiesto per installare l'applicazione completa non è molto grande, lasceremo selezionati i moduli che appaiono per default e cliccheremo Next in questa videata e in quella successiva.

È possibile cambiare la directory in cui viene installata l'applicazione anche se consigliamo di utilizzare quella che appare per default (C:\Programmi\MPLAB). Per completare l'installazione bisognerà selezionare Next nelle successive videate e, infine, riavviare il computer.

Terminata l'installazione, dato che utilizzeremo molto spesso questo programma, vi consigliamo di creare un'icona di accesso diretto sul desktop. Vi consigliamo inoltre di creare una directory per memorizzare i vostri programmi e i file generati da questi, ad esempio: C:\Programmi\MPLAB\Progetti.

Configurazione di MPLAB

A questo punto siamo già pronti per utilizzare il programma. Cliccando sull'icona di MPLAB apparirà una videata come quella riportata nella figura della pagina successiva. Dobbiamo selezionare come intendiamo lavorare su MPLAB e quale microcontroller utilizzeremo. A questo scopo selezioneremo Option dal menù superiore e, all'interno del sottomenù a tendina, selezioneremo Development mode. Qui dovremo selezionare MPLAB SIM Simulator e il processore da utilizzare, che nel nostro caso è il PIC16F870.

Se selezioniamo la cartella Clock possiamo scegliere l'oscillatore da utilizzare. Lavoreremo con un oscillatore di tipo XT a una fre-

Videata iniziale dell'installazione di MPLAB.

Moduli di MPLAB
che si possono installare.

Installeremo l'applicazione
nella directory che ci viene proposta per default.

quenza di lavoro da 4MHz. Applicheremo le variazioni con Apply e usciremo dalla configurazione accettando i cambiamenti con OK. Nella barra inferiore del programma potremo osservare le nostre modifiche.

La barra degli strumenti

Icona di accesso diretto.

MPLAB fornisce accessi rapidi alle principali funzioni mediante delle icone poste sulla barra degli strumenti. È possibile commutare quattro barre di funzioni diverse a seconda di ciò che desideriamo fare con il programma.

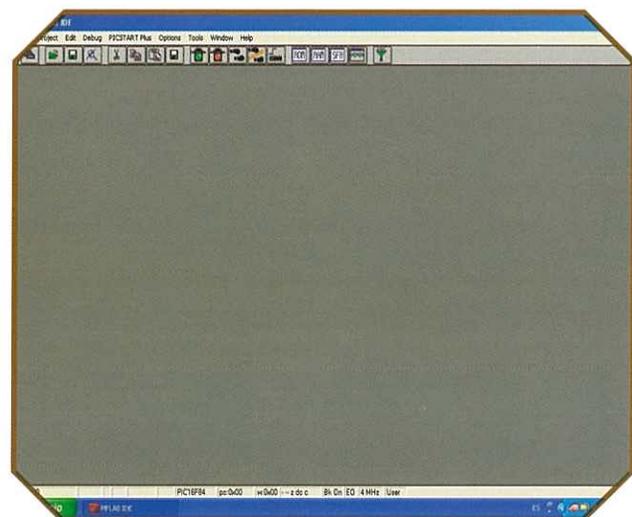
Inizio della programmazione

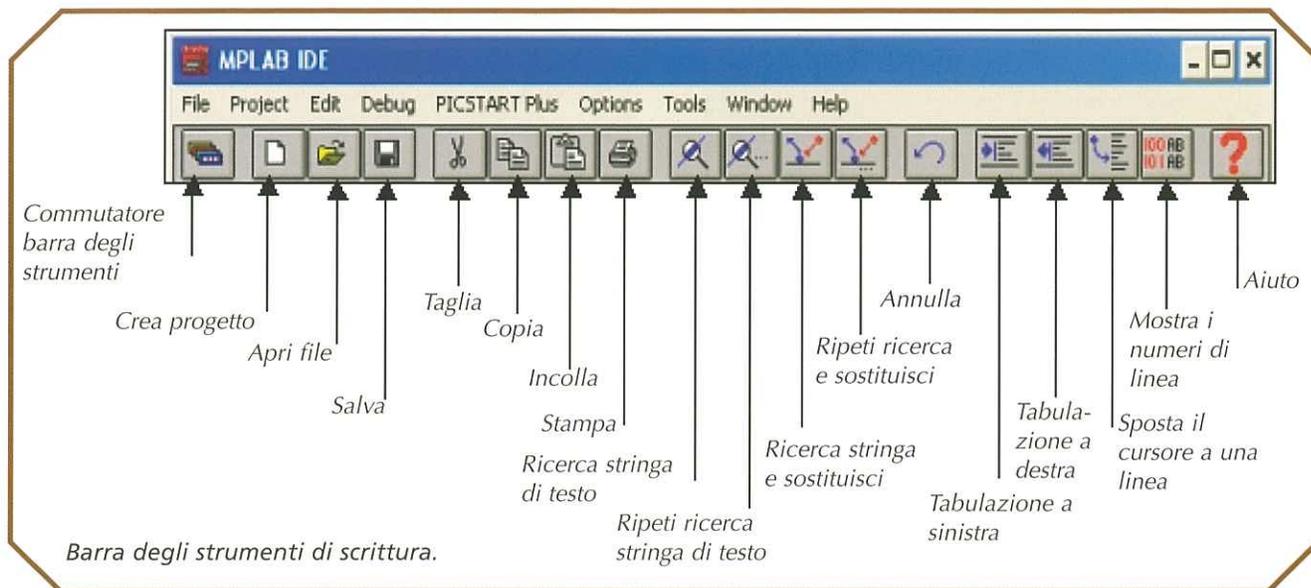
Quando vorremo sviluppare un programma, per prima cosa dovremo includerlo in un progetto. Il progetto comprenderà sia il file in assembler che progetteremo, che i file risultanti dalla compilazione.

Esistono due modi di creare un progetto, uno mediante la barra degli strumenti dove potremo scegliere l'icona adeguata, l'altro tramite i menù, selezionando File e, all'interno di questo, New. In entrambi i modi apparirà il riquadro di dialogo "Create project" mostrato nella figura.

Se rispondiamo Yes alla domanda che ci viene posta, apparirà una finestra in cui potremo assegnare un nome al progetto e scegliere la directory dove memorizzare quest'ultimo. L'indirizzo che utilizzeremo normalmente sarà: C:\Programmi\MPLAB\Progetti e il nome che daremo potrà essere uno qualsiasi, sempre che rispetti l'estensione .pjt (ad esempio: prova1.pjt).

Se clicchiamo OK apparirà il riquadro di dialogo riportato nella prima figura della pagina successiva.

Cliccando nuovamente OK potremo iniziare a creare il nostro primo programma, infatti che apparirà una videata bianca dove sarà possibile scrivere il file. Nel nostro caso, dato che ancora non sappiamo programmare, l'u-

Videata iniziale di MPLAB.

Barra degli strumenti di scrittura.

nica cosa da fare sarà copiare in questa finestra un programma progettato in precedenza.

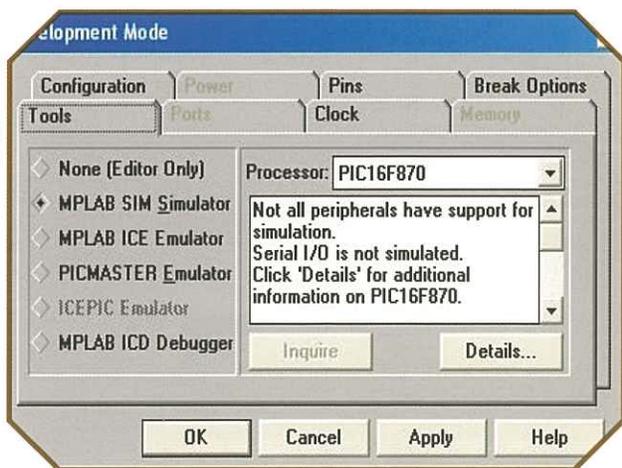
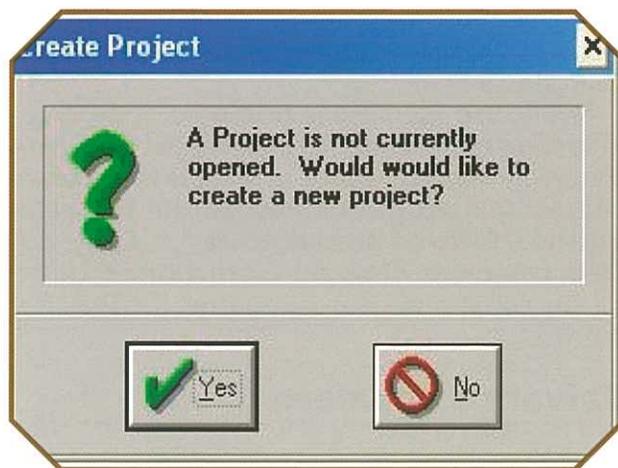
Carico dell'esempio

Sarà necessario caricare il CD che vi è stato fornito nel lettore del computer e selezionare Esercizi e Applicazioni. Aprire il primo esercizio "ese1.asm", selezionare tutto il testo e copiarlo. In seguito copiatelo sull'editor di MPLAB. La videata avrà l'aspetto della figura relativa.

Come potrete verificare la prima colonna dell'editor è riservata alle etichette. Le etichette sono espressioni alfanumeriche generate dall'utente che si associano a posizioni di memoria, è necessario iniziare da una lettera e non potranno mai essere usate espressioni utilizzate dall'assembler quali istruzioni, direttive del linguaggio, nomi di registri speciali o nomi di bit di questi registri.

Nella colonna successiva si scrive lo mnemonico delle istruzioni o direttive del linguaggio di programmazione.

Commenti di testo

È fondamentale che un programma sia commentato, in altre parole che il programmatore spieghi cosa intende fare su ogni linea di

Configurazione del programma.

Videata per la creazione di un nuovo progetto.

Primo programma con MPLAB.

Videata in cui si assegnano al progetto i file di scrittura.

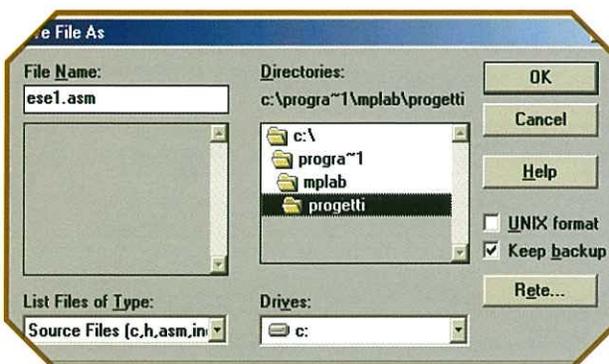
programmazione. In questo modo qualsiasi persona potrà capire facilmente il programma e potrà ampliarlo o modificarlo. Per inserire un commento è necessario porre ";" (punto e virgola) davanti al testo. Così facendo, quando il compilatore incontra questo simbolo non genererà codice macchina con ciò che segue sulla stessa linea.

Nel nostro esempio le prime tre linee di codice sono commenti. All'interno dei commenti non contano gli spazi, le linee lasciate bianche o l'utilizzo di qualsiasi parola riservata, purché il tutto sia preceduto da ";". Come potrete osservare, dopo ogni istruzione si utilizza un commento per spiegarne il funzionamento.

Salvare e chiudere

Quando avremo terminato di editare il codice in assembler (in questo caso avremo già terminato perché abbiamo copiato un programma completo) occorre salvare il file. Per questo selezioneremo File e poi Save. Apparirà il riquadro di dialogo della figura soprastante in cui dovremo assegnare un nome al file creato. Chiameremo il file "prova1.asm", ci assicureremo di aver selezionato la directory dove intendiamo salvare i nostri progetti e cliccheremo OK.

Ora potremo uscire dal programma. Selezionando la x nell'angolo superiore destro della finestra di programma, con File ed Exit il programma chiederà se vogliamo salvare i cambiamenti nel progetto, domanda alla quale dovremo rispondere Yes prima di uscire dall'applicazione.

Quando apriremo nuovamente il programma, questi ci chiederà se desideriamo aprire l'ultimo progetto memorizzato. Vi consigliamo di selezionare No e ripetere quanto imparato finora con un altro degli esercizi contenuti sul CD.

Salviamo il nostro programma.

Registri di controllo

La conoscenza dettagliata del funzionamento dei registri di controllo è fondamentale per un buon progettista. I bit di questi registri gestiscono le funzioni fondamentali e i dispositivi del processore.
Anche se non siamo pronti per analizzare la maggior parte dei registri, dato che non abbiamo ancora approfondito i dispositivi del PIC, possiamo però presentare due dei registri di controllo: STATUS, con cui dovremo familiarizzare perché ne avremo bisogno in qualsiasi programma, e PCON che – anche se non è molto importante – è ugualmente indispensabile conoscere.

Il registro di stato, STATUS

Il registro di stato occupa la quarta posizione nei quattro banchi della memoria. Al suo interno contiene lo stato delle operazioni aritmetiche della ALU, lo stato del RESET e i bit per la selezione del banco della memoria dei dati.

Se il bit è indicato come R/W (Read/Write) significa che può essere letto e scritto. Se ha solamente la R significa che è di sola lettura.

Il valore che prende il bit dopo un reset per collegamento dell'alimentazione (POR, Power On Reset) è indicato mediante '-n'.

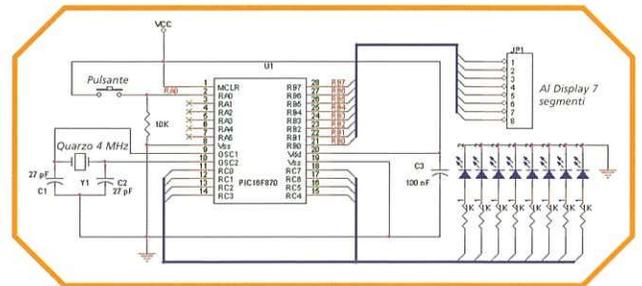
A titolo di esempio analizzeremo alcuni dei bit di questo registro:

– IRP (R/W-0): il bit può essere letto o scritto e dopo un reset per accensione prenderà il valore 0.

– C (R/W-x): il bit può essere letto o scritto e dopo un reset per accensione prenderà un valore casuale.

Quando si produce una rotazione (RRF o RLF) il bit di carry (C) si carica con il bit meno significativo o con quello più significativo, secondo il verso della rotazione.

Il registro PCON

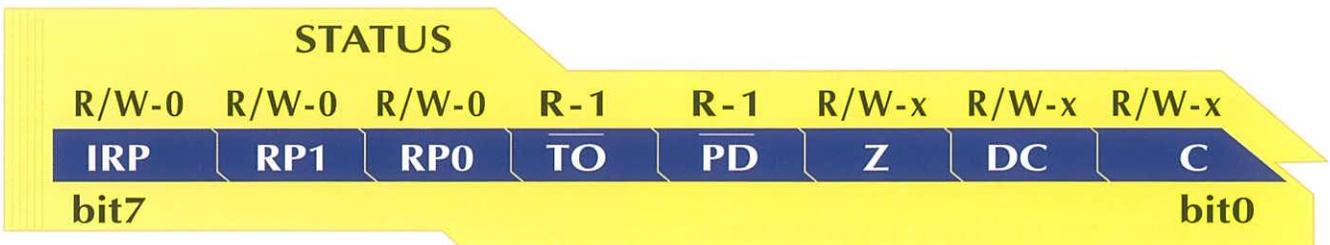
Il registro PCON si trova all'indirizzo 8Eh sul banco 1 della memoria dei dati. Contiene due bit con i quali si può differenziare se il reset è stato originato da un buco di tensione (BOR, Brown-Out) o per collegamento della stessa (POR).

Schema elettronico di un'applicazione.

Conclusioni

Abbiamo presentato due dei registri di controllo della memoria dei dati, il registro STATUS, di utilizzo comune e di grande importanza, e il registro PCON, poco utilizzato ma che dobbiamo conoscere. Il resto dei registri lo vedremo quando studieremo i dispositivi forniti dal nostro PIC16F870, lavoro che inizieremo a partire da ora.

Non è necessario memorizzare i bit che compongono i registri, infatti, quando programmiamo, potremo utilizzare questa pubblicazione come guida, dove troveremo tutti i registri spiegati bit a bit. Tuttavia man mano

STATUS

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit7							bit0

Struttura del registro di stato, STATUS.

Bit	Nome	Descrizione
Bit 7	IRP	Bit per la selezione del banco dei registri (utilizzato nell'indirizzamento indiretto)
	1	Banchi 2, 3 (100h – 1FFh)
	0	Banchi 0, 1 (00h – FFh)
Bit 6-5	RP1:RP0	Bit per la selezione del banco dei registri (utilizzato negli indirizzamenti diretti)
	11	Banco 3 (180h – 1FFh)
	10	Banco 2 (100h – 17Fh)
	01	Banco 1 (80h – FFh)
	00	Banco 0 (00h – 7Fh)
Bit 4	$\overline{TO}$	Bit di fine del tempo (Time Out)
	1	Dopo l'accensione o grazie alle istruzioni CLRWDT o SLEEP
	0	Quando termina il tempo del watchdog
Bit 3	$\overline{PD}$	Bit di spegnimento (Power Down)
	1	Dopo l'accensione o grazie all'istruzione CLRWDT
	0	Per l'esecuzione di un'istruzione SLEEP
Bit 2	Z	Bit di zero
	1	Il risultato di un'operazione aritmetica o logica è zero
	0	Il risultato di un'operazione aritmetica o logica non è zero
Bit 1	DC	Bit di Carry/Borrow sul primo digit (istruzioni ADDWF, ADDLW, SUBLW, SUBWF)
	1	Se c'è riporto sul risultato dell'operazione sui 4 bit meno significativi (digit)
	0	Se non c'è riporto sul risultato dell'operazione sui 4 bit meno significativi (digit)
Bit 0	C	Bit di Carry/Borrow (istruzioni ADDWF, ADDLW, SUBLW, SUBWF)
	1	Se c'è riporto sul risultato dell'operazione sul bit più significativo
	0	Se non c'è riporto sul risultato dell'operazione sul bit più significativo

Definizione dei bit del registro STATUS.

che avanzeremo nella preparazione dei programmi, vi renderete conto di quanto sia semplice gestire questi registri di configurazione, potendoli programmare senza la necessità di utilizzare alcun documento di appoggio.

Progetto di una applicazione

Dopo aver visto come si lavora con i registri di controllo, arriva il momento di pensare al progetto di una applicazione. Lo dobbiamo fare in modo ordinato e dobbiamo seguire sempre la stessa procedura, applicando una metodologia che ci faciliti lo sviluppo finale. Indipendentemente dal linguaggio di programmazione e dalla grandezza del progetto, il progettista deve seguire delle norme che ne facilitino l'esecuzione e la comprensione da parte dell'utente finale.

Definizione del problema

Il primo passo per realizzare un progetto è aver chiaro ciò che vogliamo fare. Il sistema da sviluppare dovrà soddisfare una serie di requisiti e di specifiche.

Quindi la prima cosa che dovrà fare un progettista quando affronta un nuovo progetto sarà l'analisi del problema.

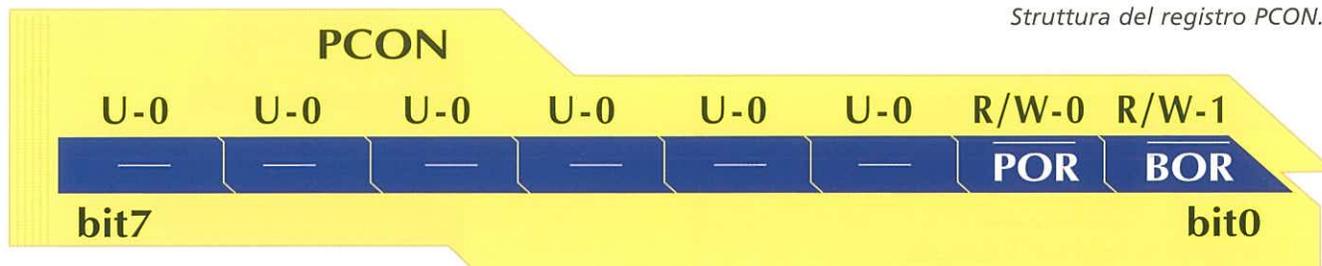
Schema elettronico

Abbiamo enunciato i punti principali che caratterizzano il nostro progetto, sappiamo ciò che vogliamo fare, ora però dobbiamo chiederci come farlo.

In molti sistemi sarà necessario implementare un circuito elettronico che soddisfi le necessità del nostro progetto.

Definizione del problema:
- Voglio visualizzare in sequenza i numeri da 0 a 9.
- Voglio che quando lo desidero appaia visualizzato un numero casuale.
- Voglio che questa visualizzazione duri un tempo preciso dopodiché si ripeta la visualizzazione sequenziale.

Dobbiamo avere ben chiaro ciò che vogliamo fare.

Struttura del registro PCON.

Nel nostro caso abbiamo bisogno di visualizzare un numero, a questo scopo, useremo un display a 7 segmenti e/o dei diodi LED per la rappresentazione binaria. Avremo bisogno anche di un pulsante per determinare il momento in cui desideriamo che si visualizzi il numero casuale e ovviamente il PIC, con il suo oscillatore e la sua alimentazione.

Tutte queste idee si plasmano in un circuito elettronico in cui il PIC è la parte centrale dove saranno collegati tutti gli elementi.

Gli schemi elettronici possono essere più o meno complessi secondo l'applicazione e possono essere fatti a mano oppure utilizzando un software specifico (ORCAD, Electronics Workbench, ecc.).

Lo schema elettronico ci permetterà di definire una soluzione al problema iniziale e determinare l'hardware necessario per lo sviluppo, oltre alla configurazione che dovremo programmare sul microcontroller.

Organigrammi

Un organigramma è la rappresentazione grafica di ciò che deve fare il programma. Dobbiamo creare una traccia che ci permetta di seguire un certo ordine durante la programmazione, e questo lo otterremo creando precedentemente un organigramma.

Un organigramma si rappresenta mediante:

– Rettangoli, che contengono le azioni da eseguire.

– Rombi, dove si specificano le condizioni che determinano il percorso da seguire.

– Frecce, che indicano l'ordine seguito dalle istruzioni. Se fuoriescono da un rombo devono essere etichettate per scegliere l'alternativa correttamente.

Gli organigrammi avranno un unico punto di ingresso e un unico punto di uscita.

Normalmente per un'applicazione complessa si creano diversi organigrammi. Esisterà sempre l'organigramma generale dell'applicazione, però conviene realizzare organigrammi dettagliati di ogni parte o modulo che compone l'applicazione.

Come potrete verificare, ciò che in principio era solamente un'idea sta già prendendo forma. Si è concretizzata, è stata definita e possiamo già iniziare a programmare.

Programma

Quando affrontiamo la programmazione conviene tenere lo schema elettronico e gli orga-

Bit 7.2		**Non implementati (Si leggono come 0)**
Bit 1	**POR**	**Flag di "Power On Reset"**
	1	**Non c'è stato un reset per collegamento dell'alimentazione**
	0	**C'è stato un reset per collegamento dell'alimentazione (deve essere reimpostato a 1 via software)**
Bit 0	**BOR**	**Flag di "Brown-out Reset"**
	1	**Non si è verificato un abbassamento di tensione**
	0	**Si è verificato un abbassamento di tensione (dovrà essere reimpostato a 1 via software)**

Definizioni dei bit del registro PCON.

Possiamo verificare il corretto funzionamento del programma mediante un simulatore.

Organigramma generale di un'applicazione.

nigrammi a portata di mano. La programmazione non è altro che la scrittura delle nostre idee in un linguaggio comprensibile alla macchina. Se partiamo da organigrammi dettagliati e ben fatti non sarà molto difficile tradurli in istruzioni. Con lo schema elettronico potremo configurare il microcontroller, dato che conosceremo tutti i suoi requisiti hardware.

È molto importante commentare il programma mediante spiegazioni di ciò che fanno le istruzioni. Così il programma potrà essere capito e messo a punto dopo la sua realizzazione.

Messa a punto

Dopo aver terminato il programma, dobbiamo compilarlo prima di eseguirlo. È molto difficile, anche per un esperto programmatore, che un programma funzioni bene al primo colpo. Nella fase di compilazione si potranno correggere gli errori di sintassi, prodotti per assenza di parametri, parole scritte male, ecc. Il programma non verrà compilato fino a quando non saranno stati corretti gli errori.

Fatto questo, occorrerà verificare che il programma realizzato esegua realmente ciò che vogliamo. Prima di montare il sistema fisico finale conviene analizzare il funzionamento del programma su un simulatore (MPLAB), potremo così valutare il funzionamento di ogni istruzione e del programma nel suo insieme.

Infine scriveremo il programma sul microcontroller e monteremo il circuito, pronti a far fronte agli eventuali errori che ci potrebbero essere durante lo sviluppo hardware del sistema.